```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <time.h>
#define Fsize 3
typedef struct packet
{
        int SeqNum;
        char Data[Fsize+1];
}PACKET;
PACKET *Data_to_send, *Data_received;
time_t t ;
// Breaks the message into packets
int divide(char *msg)
{
        int msglen, NoOfPacket, i, j;
        msglen = strlen(msg);
        NoOfPacket = msglen/Fsize;
if((msglen%Fsize)!=0)
                NoOfPacket++;
Data_to_send = (PACKET *)malloc(sizeof(PACKET) *NoOfPacket);
        for(i = 0; i < NoOfPacket; i++)
        {
                Data_to_send[i].SeqNum = i + 1;
                for (j = 0; (j < Fsize) && (*msg != '\0'); j++, msg++)
                        Data_to_send[i].Data[j] = *msg;
                Data_to_send[i].Data[j] = '\0';
        }

        printf("\nThe Message has been divided as follows\n");
        printf("\nPacket No.\tData\n\n");
```

```
	for (i = 0; i < NoOfPacket; i++)
		printf(" %d\t\t%s\n", Data_to_send[i].SeqNum, Data_to_send[i].Data);

	return NoOfPacket;
}
// shuffles the packets
void shuffle(int NoOfPacket)
{
	int *DisOrder;
	int i, j, trans;
	srand(time(&t)); //every time you shuffle,get different random sequence
	DisOrder=(int * )calloc(NoOfPacket, sizeof(int));
	Data_received = (PACKET *)malloc(sizeof(PACKET) * NoOfPacket);
	for (i = 0; i < NoOfPacket;)
	{
		trans = rand()%NoOfPacket;
		if (DisOrder[trans]!=1)
		{
			Data_received[i].SeqNum = Data_to_send[trans].SeqNum;
			strcpy(Data_received[i].Data, Data_to_send[trans].Data);
			i++;
			DisOrder[trans] = 1;
		}
	}

	free(DisOrder);
}
// sorts the packets
void sortframes(int NoOfPacket)
{
	int i, j;
```

```
  PACKET temp;
for (i = 0; i < NoOfPacket; i++)
        {
                for (j = 0; j < NoOfPacket - (i+1); j++)
                {
                        if (Data_received[j].SeqNum > Data_received[j + 1].SeqNum)
                        {
                                temp.SeqNum = Data_received[j].SeqNum;
                                strcpy(temp.Data, Data_received[j].Data);
                                Data_received[j].SeqNum = Data_received[j + 1].SeqNum;
                                strcpy(Data_received[j].Data, Data_received[j + 1].Data);
                                Data_received[j + 1].SeqNum = temp.SeqNum;
                                strcpy(Data_received[j + 1].Data, temp.Data);
                        }
                }
        }
}

// receives packets out of order and calls sort function
void receive(int NoOfPacket)
{
        int i, j;
    PACKET temp;
        printf("\nPackets received in the following order\n");
        for (i = 0; i < NoOfPacket; i++)
                printf("%4d",Data_received[i].SeqNum);
       sortframes(NoOfPacket);
        printf("\n\nPackets in order after sorting..\n");
        for (i = 0; i < NoOfPacket; i++)
                printf("%4d",Data_received[i].SeqNum);
        printf("\n\nMessage received is :\n");
```

```
        for (i = 0; i < NoOfPacket; i++)
                printf("%s",Data_received[i].Data);
}

int main()
{
        char msg[25];
        int NoOfPacket;
        printf("\nEnter The message to be Transmitted :\n");
        scanf("%[^\n]", msg);
        NoOfPacket = divide(msg);
        shuffle(NoOfPacket);
        receive(NoOfPacket);
        free(Data_to_send);
        free(Data_received);
        return 0;
}
```